**03. Streaming Service Selector**

**Your Task**

Using **Mocha** and **Chai** write **JS Unit Tests** to test a variable named **streamingServiceSelector**, which represents an object. You may use the following code as a template:

|  |
| --- |
| describe(**"*Tests* …"**, **function**() {  describe(**"*TODO* …"**, **function**() {  ***it***(**"*TODO …*"**, **function**() {  *//* ***TODO:*** …  });  });  *//* ***TODO:*** …  }); |

The object should have the following functionality:

* **selectingContent(type, platform, genre) -** A function that accepts **three** parameters:
  + **type**: **string** (either "Movie" or "TV Show"),
  + **platform**: **string** (name of the streaming platform, e.g., "Netflix"),
  + **genre**: **string** (type of genre, e.g., "Action", "Comedy", etc.).
* If the value of **genre** is not in the list of supported genres (["Action", "Comedy", "Drama", "Thriller", "Horror", "Romance", "Sci-Fi"]), **throw** an error:

**"We currently support these genres: Action, Comedy, Drama, Thriller, Horror, Romance, Sci-Fi."**

* If the value of **type** is different from "**Movie**" or "**TV Show**", **throw** an error:

**"We currently only support 'Movie' or 'TV Show' types."**

* If all inputs are **valid**, **return** the following message:

**"You can watch this {genre} {type} on {platform}. Enjoy your {genre}-filled experience!"**

* **availablePlatforms(platforms,** **selectedPlatformIndex) -** A function that accepts an **array** and **number**. The **platforms** array will store the brand names **(["Netflix", "HBO", "Disney+"])** and **selectedPlatformIndex** is an integer that specifies the index of the currently selected platform**.**
  + You must **remove** the **platform** located at the **selectedPlatformIndex** from the array.
  + Finally, **return** the following string:

"Other available platforms are: {platform}, …, {platform}."

* + There is a **need for validation** of the input, the **array** and index may not always be valid. In case of submitted **invalid** parameters, **throw** an error **"Invalid platform selection."**
    - If passed **platforms** parameteris **not** an array.
    - If the **selectedPlatformIndex** is not a number and is outside the limits of the array.
* **contentRating(runtimeInMinutes, viewerRating) -** A function that accepts two parameters:
  + **runtimeInMinutes**: a **number** representing the runtime of the content in minutes.
  + **viewerRating**: a **number** representing the viewer's rating of the content (range: 0 to 10).
* Convert **runtimeInMinutes** to hours (formatted to two decimal places) and validate both inputs.
* If **viewerRating** is **greater** than or **equal** to 7, **return** the following message:

**"This content is highly rated ({viewerRating}/10) and has a runtime of {runtimeInHours} hours. Enjoy your watch!"**

* If **viewerRating** is **less** than 7, **return** the following message:

**"This content has a lower rating ({viewerRating}/10) and runs for {runtimeInHours} hours. You might want to check reviews first."**

* You **need to validate** the input, if **runtimeInMinutes** is **not** a positive number, or **viewerRating** is **not** a number between 0 and 10, **throw** an error: **"Invalid runtime or rating."**

**JS Code**

To ease you in the process, you are provided with an implementation that meets all of the specification requirements for the **streamingServiceSelector** object:

|  |
| --- |
| streamingServiceSelector.js |
| const streamingServiceSelector = {      selectingContent(type, platform, genre) {          const supportedGenres = ["Action", "Comedy", "Drama", "Thriller", "Horror", "Romance", "Sci-Fi"];          if (!supportedGenres.includes(genre)) {              throw new Error(`We currently support these genres: ${supportedGenres.join(", ")}.`);          }          if (type !== "Movie" && type !== "TV Show") {              throw new Error(`We currently only support 'Movie' or 'TV Show' types.`);          }          return `You can watch this ${genre} ${type} on ${platform}. Enjoy your ${genre}-filled experience!`;      },      availablePlatforms(platforms, selectedPlatformIndex) {          if (!Array.isArray(platforms) || !Number.isInteger(selectedPlatformIndex) ||              selectedPlatformIndex < 0 || selectedPlatformIndex >= platforms.length) {              throw new Error("Invalid platform selection.");          }          let available = platforms.filter((\_, index) => index !== selectedPlatformIndex);          return `Other available platforms are: ${available.join(", ")}.`;      },      contentRating(runtimeInMinutes, viewerRating) {          let runtimeInHours = (runtimeInMinutes / 60).toFixed(2);          if (typeof runtimeInMinutes !== "number" || runtimeInMinutes <= 0 ||              typeof viewerRating !== "number" || viewerRating < 0 || viewerRating > 10) {              throw new Error("Invalid runtime or rating.");          }          if (viewerRating >= 7) {              return `This content is highly rated (${viewerRating}/10) and has a runtime of ${runtimeInHours} hours. Enjoy your watch!`;          } else {              return `This content has a lower rating (${viewerRating}/10) and runs for ${runtimeInHours} hours. You might want to check reviews first.`;          }      }  } |

**Submission**

Submit your tests inside a **describe()** statement, as shown above.